Due Date: June 29, 2004

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re Application of:

Inventor: Neelakantan Sundaresan

Serial #: 09/191,281

Filed: November 12, 1998

Title: GENERATING VISUAL EDITORS FROM SCHEMA DESCRIPTIONS

Examiner: Rachna Singh

Group Art Unit: 2176

Appeal No.: _____

BRIEF OF APPELLANT

MAIL STOP APPEAL BRIEF - PATENTS
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

In accordance with 37 CFR §1.192, Appellant's attorney hereby submits the Brief of Appellant, in triplicate, on appeal from the final rejection in the above-identified application as set forth in the Office Action dated January 30, 2004.

Please charge the amount of $330.00 to cover the required fee for filing this Brief as set forth under 37 CFR §1.17(c) to Deposit Account No. 09-0441 of IBM Corporation, the assignee of the present application. Also, please charge any additional fees or credit any overpayments to Deposit Account No. 09-0441.

I. REAL PARTY IN INTEREST

The real party in interest is IBM Corporation, the assignee of the present application.

II. RELATED APPEALS AND INTERFERENCES

There are no related appeals or interferences for the above-referenced patent application.

III. STATUS OF CLAIMS

Claims 1-69 are pending in the application.

Claims 1, 4-7, 24, 27-30, 47, and 50-53 were rejected under 35 U.S.C. §103(a) as being unpatentable over Dale, U.S. Patent No. 6,272,673 (Dale) in view of Kyojima et al., U.S. Patent No. 5,920,879 (Kyojima).

Claims 2, 17-18, 25, 40-41, 48, and 63-64 were rejected under 35 U.S.C. §103(a) as being unpatentable over Dale in view of Kyojima, as applied to claims 1, 24, and 47, and further in view of Bray, "Extensible Markup Language (XML): Part I. Syntax," http://www.w3.org/TR/WD-xml-lang-970331 (Bray).

Claims 3, 26, and 49 were rejected under 35 U.S.C. §103(a) as being unpatentable over Dale in view of Kyojima and Bray as applied to claims 2, 25, and 48, and further in view of Kirsanov, "XML DTDs and Valid XML Documents," http://www.webreference.com/dlab/books/html/38-3.html (Kirsanov) and Bray, "Document Content Description for XML," http://www.w3.org/TR/NOTE-dcd, (Bray 2).

Claims 8-10, 21, 31-33, 44-45, 54-56, and 67-68 were rejected under 35 U.S.C. §103(a) as being unpatentable over Dale in view of Kyojima as applied to claims 1, 24, and 47, and further in view of Softquad HotMetalPro 3.0 User's Manual, 1996, pages 77-83 (HotMetalPro).

Claims 11-15, 34-38, and 57-61 were rejected under 35 U.S.C. §103(a) as being unpatentable over Dale, Kyojima and HotMetalPro as applied to claims 10, 33, and 56, and further in view of W3C Extensible Markup Language (XML) 1.0, 2/1998, available: http://www.w3.org/TR/1998/REC-xml-19980210 (W3C).

Claims 16, 23, 34-37, 39, 46, 57-60, 62, and 69 were rejected under 35 U.S.C. §103(a) as being unpatentable over Dale in view of Kyojima as applied to claims 1, 24, and 47, and further in view of W3C.

Claims 19-20, 42-43, and 65-66 were rejected under 35 U.S.C. §103(a) as being unpatentable over Dale in view of Kyojima and Bray as applied to claim 18, and further in view of HotMetalPro.

IV. STATUS OF AMENDMENTS

No amendments to the claims have been made subsequent to the final Office Action.

V. SUMMARY OF THE INVENTION

Briefly, Appellant's invention, as recited in independent claims 1, 24 and 47, is generally directed to generating a document editor. Claim 1 is representative and comprises:

(a) generating one or more class specifications in the computer from a schema for the document, wherein the class specifications identify user interface components of the editor corresponding to entities defined in the schema; and

(b) instantiating one or more objects in the computer from the class specifications to invoke the editor.

With regard to the rejected claims, refer to the specification as follows:

(a) at page 8, line 9 through page 8, line 18;

(b) at page 9, line 1 through page 10, line 13 and in FIG. 1 as reference numbers 100-118; and

(c) at page 10, line 15 through page 26, line 8 and in FIGS. 1-7 as reference numbers 100-118 and 200-206.

VI. ISSUES PRESENTED FOR REVIEW

1. Whether claims 1, 4-7, 24, 27-30, 47, and 50-53 are obvious under 35 U.S.C. §103(a) over Dale, U.S. Patent No. 6,272,673 (Dale) in view of Kyojima et al., U.S. Patent No. 5,920,879 (Kyojima).

2. Whether claims 2, 17-18, 25, 40-41, 48, and 63-64 are obvious under 35 U.S.C. §103(a) over Dale in view of Kyojima, as applied to claims 1, 24, and 47, and further in view of Bray, "Extensible Markup Language (XML): Part I. Syntax," http://www.w3.org/TR/WD-xml-lang-970331 (Bray).

3. Whether claims 3, 26, and 49 are obvious under 35 U.S.C. §103(a) over Dale in view of Kyojima and Bray as applied to claims 2, 25, and 48, and further in view of Kirsanov, "XML DTDs and Valid XML Documents," http://www.webreference.com/dlab/books/html/38-3.html (Kirsanov) and Bray, "Document Content Description for XML," http://www.w3.org/TR/NOTE-dcd, (Bray 2).

4. Whether claims 8-10, 21, 31-33, 44-45, 54-56, and 67-68 are obvious under 35 U.S.C. §103(a) over Dale in view of Kyojima as applied to claims 1, 24, and 47, and further in view of Softquad HotMetalPro 3.0 User's Manual, 1996, pages 77-83 (HotMetalPro).

5. Whether claims 11-15, 34-38, and 57-61 are obvious under 35 U.S.C. §103(a) over Dale, Kyojima and HotMetalPro as applied to claims 10, 33, and 56, and further in view of W3C Extensible markup Language (XML) 1.0, 2/1998, available: http://www.w3.org/TR/1998/REC-xml-19980210 (W3C).

6. Whether claims 16, 23, 34-37, 39, 46, 57-60, 62, and 69 are obvious under 35 U.S.C. §103(a) over Dale in view of Kyojima as applied to claims 1, 24, and 47, and further in view of W3C.

7. Whether claims 19-20, 42-43, and 65-66 are obvious under 35 U.S.C. §103(a) over Dale in view of Kyojima and Bray as applied to claim 18, and further in view of HotMetalPro.

VII. GROUPING OF CLAIMS

The rejected claims do not stand or fall together. Each of the following claim groups is independently patentable:

(a) claims 1, 24 and 47 stand or fall together;

(b) claims 2, 25 and 48 stand or fall together;

(c) claims 3, 26 and 49 stand or fall together;

(d) claims 4, 27 and 50 stand or fall together;

(e) claims 5, 28 and 51 stand or fall together;

(f) claims 6, 29 and 52 stand or fall together;

(g) claims 7, 30 and 53 stand or fall together;

(h) claims 8, 31 and 54 stand or fall together;

(i) claims 9, 32 and 55 stand or fall together;

(j) claims 10, 33 and 56 stand or fall together;

(k) claims 11, 34 and 57 stand or fall together;

(l) claims 12, 35 and 58 stand or fall together;

(m) claims 13, 36 and 59 stand or fall together;

(n) claims 14, 37 and 60 stand or fall together;

(o) claims 15, 38 and 61 stand or fall together;

(p) claims 16, 39 and 62 stand or fall together;

(q) claims 17, 40 and 63 stand or fall together;

(r) claims 18, 41 and 64 stand or fall together;

(s) claims 19, 42 and 65 stand or fall together;

(t) claims 20, 43 and 66 stand or fall together;

(u) claims 21, 44 and 67 stand or fall together;

(v) claims 22, 45 and 68 stand or fall together; and

(w) claims 23, 46 and 69 stand or fall together.

Separate arguments for each of the claim groups are provided below.

VIII. ARGUMENTS

A. The Office Action Rejections

In the Office Action, claims 1, 4-7, 24, 27-30, 47, and 50-53 are rejected under 35 U.S.C. §103(a) as being unpatentable over Dale, U.S. Patent No. 6,272,673 B1 (Dale) in view of Kyojima et al., U.S. Patent No. 5,920,879 (Kyojima); claims 2, 17-18, 25, 40-41, 48, and 63-64 are rejected under 35 U.S.C. §103 as being unpatentable over Dale in view of Kyojima, as applied to claims 1, 24, and 47, and further in view of Bray, Extensible Markup Language (XML): Part I. Syntax, (Bray); claims 3, 26, and 49 are rejected under 35 U.S.C. §103(a) as being unpatentable over Dale in view of Kyojima and Bray as applied to claims 2, 25, and 48, and further in view of Kirsanov, XML DTDs and Valid XML Documents (Kirsanov) and Bray, Document Content Description for XML (Bray 2); claims 8-10, 21, 31-33, 44-45, 54-56, and 67-68 are rejected under 35 U.S.C. §103(a) as being unpatentable over Dale in view of Kyojima as applied to claims 1, 24, and 47, and further in view of Softquad HotMetalPro 3.0 User's Manual (HotMetalPro); claims 11-15, 34-38, and 57-61 are rejected under 35 U.S.C. §103(a) as being unpatentable over Dale, Kyojima, HotMetalPro as applied to claims 10, 33, and 56, and further in view of W3C Extensible Markup Language – XML, 1.0 (W3C); claims 16, 23, 34-37, 39, 46, 57-60, 62, and 69 are rejected under 35 U.S.C. §103(a) as being unpatentable over Dale in view of Kyojima as applied to claims 1, 24, and 47, and further in view of W3C; and claims 19-20, 42-43, and 65-66 are rejected under 35 U.S.C. §103(a) as being unpatentable over Dale in view of Kyojima and Bray as applied to claim 18, and further in view of HotMetalPro.

Appellant's attorney traverses these rejections.

B. Appellant's Independent Claims

Independent claims 1, 24 and 47 are generally directed to generating a document editor. Claim 1 is representative and comprises:

(a) generating one or more class specifications in the computer from a schema for the document, wherein the class specifications identify user interface components of the editor corresponding to entities defined in the schema; and

(b) instantiating one or more objects in the computer from the class specifications to invoke the editor.

C. The Dale Reference

Dale describes a mechanism for automatically establishing connections between executable components of a hypertext-based application is provided. An application created using the mechanism includes a plurality of hypertext-based pages, at least some of which incorporate executable components. The application is invoked by a hypertext request for a page from a browser running on a client tier. An application server responds to the request by retrieving the requested page and assigning any components incorporated therein to the proper tier for execution. The mechanism provides a single model by which any executable component can be specified by an application developer for execution on any tier on the network or made subject to an automatic, dynamic tier assignment by the application server. Components of a given application can be distributed across, and specified for execution on, three or more different tiers and moved from tier to tier. An application developer can use a conventional hypertext editor to integrate selected components into extended hypertext pages to create an application.

D. The Kyojima Reference

Kyojima describes a document structure composing apparatus to judge a short part based upon the structure of the whole document, perform processing for complementation and compose document structure according to a desired document class. Complementation specification storage means stores the specification of complementation for document structure. Complementation means applies processing for complementation to pre-complemented document structure.

Correlating rule storage means stores a rule for correlating components between different document classes. Document structure converting means converts an original document which meets the structural constraint of a specific document class to structure according to another document class according to the correlating rule. The original document is first converted from structure according to the specific document class to structure nearly according to a desired document class by the document structure converting means. The correlating rule required for conversion is stored in the correlating rule storage means. Afterward, processing for complementation based upon the specification of complementation stored in the complementation specification storage means is executed by the complementation means and document structure according to a desired document class is composed.

E. The Bray Reference

Bray is a manual for the Extensible Markup Language (XML), which is a dialect of Standard Generalized Markup Language (SGML). XML describes a class of data objects called XML documents, and partially describes the behavior of computer programs which process them. XML documents are made up of storage units called entities, which contain either text or binary data. Text is made up of characters, some of which form the character data in the document, and some of which form markup. Markup encodes a description of the document's storage layout and logical structure. XML provides a mechanism to impose constraints on the storage layout and logical structure. A software module called an XML processor is used to read XML documents and provide access to their content and structure. It is assumed that an XML processor is doing its work on behalf of another module, referred to as the application. This specification describes the required behavior of an XML processor in terms of how it must read XML data and the information it must provide to the application.

F. The Kirsanov Reference

Kirsanov describes XML DTDs and valid XML documents. A DTD allows an XML parser to validate a document. When validating, the parser checks for misspelled tags or attributes, for errors in types of attribute values and in elements' content models, and so on. For HTML, similar validation services exist that will check a file against one of the existing HTML DTDs. In addition,

for human readers, a DTD is a convenient way to quickly learn the structure of the particular type of documents. Compared to SGML, the simplified DTD syntax of XML is very straightforward and unambiguous. Finally, with DTDs, the user can define not only elements and their attributes, but also entities. Similarly to macros in word processors or #define preprocessor instructions in C, entities can be used to abbreviate text strings and markup instructions in an obvious and easy-to-modify manner. Also, external entities can be used to refer to other XML documents, DTDs, or binary data located in separate files.

G. The Bray2 Reference

Bray2 is a manual for the Document Content Description (DCD), which specifies rules covering the structure and content of XML documents. DCD is expressed in a way that is consistent with the W3C RDF (Resource Description Framework) effort. In particular, DCD is an RDF vocabulary. DCD is intended to define document constraints in an XML syntax; these constraints may be used in the same fashion as traditional XML DTDs. DCD also provides additional properties, such as basic datatypes.

H. The HotMetalPro Reference

HotMetalPro is a user manual for Softquad HotMetalPro 3.0, dated 1996. Pages 77-83 are included, which describe

I. The W3C Reference

W3C is a manual for the Extensible Markup Language (XML).

J. Appellant's Independent Claims Are Patentable Over The References

Appellant's independent claims 1, 24 and 47 are patentable over the references because they recite a novel and nonobvious combination of elements. None of the references, taken individually or in any combination, teaches or suggests this combination of elements.

The Office Action cites Dale as teaching the limitations of Appellant's independent claims directed to "generating one or more class specifications in the computer," at col. 10, lines 50-59. However, the complete limitation of Appellant's independent claims recites "generating one or more

class specifications in the computer from a schema for the document, wherein the class specifications identify user interface components of the editor corresponding to entities defined in the schema," and the complete limitation is not shown by Dale. In contrast, the editor in Dale merely constructs web-based applications, wherein the web pages include tags to applets that may be downloaded when the pages are accessed, and the director component is merely an applet interconnects other components (applets).

The Office Action also cites Dale as teaching displaying a graphical user interface and using an editor to integrate components into a hypertext page to create an application, and that Dale's editor allows text to be combined with various components, at the Abstract and columns 19-20. However, as noted above, the complete limitation of Appellant's independent claims recites "generating one or more class specifications in the computer from a schema for the document, wherein the class specifications identify user interface components of the editor corresponding to entities defined in the schema," and the complete limitation is not shown by Dale. In contrast, Dale's editor, which is used to construct web-based applications including web pages having references or tags to applets, is not generated from class specifications that are themselves generated from a schema for the document, wherein the class specifications identify user interface components of the editor corresponding to entities defined in the schema.

In addition, the Office Action cites Dale as teaching "instantiating one or more objects in the computer from the class specifications," at col. 10, lines 60-63, as Dale describes instantiating components, particularly Java components, at col. 5, lines 7-9. However, the complete limitation of Appellant's independent claims recites "instantiating one or more objects in the computer from the class specifications to invoke the editor," in the context where the class specifications are generated from a schema for the document and identify user interface components of the editor corresponding to entities defined in the schema, and the complete limitation is not shown by Dale. In contrast, the components described in Dale comprise objects instantiated from some unspecified class specifications, but they are not class specifications generated from a schema for the document that identify user interface components of the editor corresponding to entities defined in the schema, wherein the components or objects described in Dale are invoked by the web pages, and do not themselves invoke the editor that constructs the web pages.

The Office Action admits that Dale does not explicitly state that the components are class specifications, but cites Kyojima, at the Abstract and cols. 1-4, as disclosing the generation of "class specifications" from "schemas." Appellant's attorney disagrees. Nowhere does Kyojima teach or suggest class specifications generated from a schema for a document, in the context where the class specifications identify user interface components of the editor corresponding to entities defined in the schema and are instantiated as objects to invoke the editor. Instead, Kyojima merely describes document structure conversion, wherein the term class is used to refer to a type of document structure.

The Office Action asserts that that Dale teaches the limitations "wherein the class specifications identify user interface components of the editor corresponding to entities defined in the schema," when it discloses an application that is invoked in response to a request from a client, for a page that includes references ("tags") to one or more components, wherein the components are embodied as Java classes, and that Dale teaches web-based development tools that provide functional connections between components of a web application to allow components to communicate with each other, wherein all components have parameters, at cols. 5-6. The Office Action also asserts that Dale teaches using Java classes to identify user interface components of an editor, as discussed in cols. 5-6 and as shown in FIG.11. Appellant's attorney disagrees. Nowhere does Dale teach or suggest class specifications generated from a schema for a document, in the context where the class specifications identify user interface components of the editor corresponding to entities defined in the schema and are instantiated as objects to invoke the editor. Dale merely describes components that are applets, which are embodied as Java classes and embedded in HTML pages.

Finally, the Office Action asserts that that Kyojima teaches the limitations "generating class specifications from schemas," when it discloses that data type definitions (schema) play a role in document class where the structured document must comply with the constraints of a document class, at col. 1. Moreover, the Office Action asserts that it would have been obvious to combine Dale and Kyojima as components models would adapt to other component models according to structural constraints, at col. 3 of Kyojima. Appellant's attorney disagrees. Nowhere does Kyojima teach or suggest class specifications generated from a schema for a document, in the context where the class specifications identify user interface components of the editor corresponding to entities

defined in the schema and are instantiated as objects to invoke the editor. Kyojima merely describes document structure conversion, wherein the term class is used to refer to a type of document structure.

Consequently, Appellant's attorney respectfully submits that the combination of Dale and Kyojima does not render obvious the limitations of Appellant's independent claims. Indeed, it would only be with hindsight to assert that the specific limitations of Appellant's claims are rendered obvious by this combination.

Moreover, the other references fail to overcome the deficiencies of Dale and Kyojima. Recall that the other references were only cited against dependent claims 2-3, 8-23, 25-26, 31-46, 48-49 and 54-69, in combination with Dale and Kyojima.

Appellant's attorney asserts that the differences between Appellant's claimed invention and the cited references result in operational advantages of the Appellant's invention over the cited references. In addition, Appellant's invention solves problems not recognized by the cited references.

Thus, Appellant's attorney submits that independent claims 1, 24 and 47 are allowable over the cited references.

K. Appellant's Dependent Claims Are Patentable Over The References

Dependent claims 2-23, 25-46 and 48-69 are submitted to be allowable over the cited references in the same manner, because they are dependent on independent claims 1, 24 and 47, respectively, and thus contain all the limitations of the independent claims. In addition, dependent claims 2-23, 25-46 and 48-69 recite additional novel elements not shown by the cited references.

With regard to claims 2, 25 and 48, which recite that the documents are eXtensible Markup Language (XML) documents and the schemas are XML schemas, the Office Action asserts that these limitations are taught by Bray at page 7+. Appellant's attorney disagrees. At the indicated location, Bray merely describes XML documents and schemas, but does not teach or suggest the use of XML documents and schemas in the same context as Appellant's claimed invention.

With regard to claims 3, 26 and 49, which recite that the schemas are selected from a group including Document Type Definition (DTD) schemas, Document Content Definition (DCD) schemas, and XSchema schemas, the Office Action asserts that these limitations are taught by

Kirsanov at pages 1-3 and Bray 2 at page 2. Appellant's attorney disagrees. At the indicated location, Kirsanov and Bray 2 merely describe Document Type Definition (DTD) schemas, Document Content Definition (DCD) schemas, and XSchema schemas, but does not teach or suggest Document Type Definition (DTD) schemas, Document Content Definition (DCD) schemas, and XSchema schemas in the same context as Appellant's claimed invention.

With regard to claims 4, 27 and 50, which recite that the class specifications comprise Java class specifications, the Office Action asserts that these limitations are taught by Dale at col. 13, lines 1-3. Appellant's attorney disagrees. At the indicated location, Dale merely describes Java class specifications, but does not teach or suggest Java class specifications in the same context as Appellant's claimed invention.

With regard to claims 5, 28 and 51, which recite that the generating further comprises converting an entity defined in the schema into the class specification, the Office Action asserts that these limitations are taught by Dale at col. 3, lines 64-67. Appellant's attorney disagrees. At the indicated location, Dale merely describes the execution of instructions by a CPU, but does not teach or suggest the above-identified limitations of Appellant's claims.

With regard to claims 6, 29 and 52, which recite that the generating further comprises generating the class specifications in the computer from the schemas and one or more optional customization specifications, the Office Action asserts that these limitations are taught by Dale at col. 20, lines 5-15. Appellant's attorney disagrees. At the indicated location, Dale merely describes an HTML page with executable components, but does not teach or suggest the above-identified limitations of Appellant's claims.

With regard to claims 7, 30 and 53, which recite that the optional customization specifications define what class names to generate for each entity defined in the schema, the Office Action asserts that these limitations are taught by Dale at col. 14, lines 57-60. Appellant's attorney disagrees. At the indicated location, Dale merely describes an applet definition, but does not teach or suggest similar customization specifications.

With regard to claims 8, 31 and 54, which recite that the class specifications include one or more specifications selected from a group comprising (1) a visual editor class specification, (2) a content implementation class specification, and a handler class specification, the Office Action asserts that these limitations are taught by HotMetalPro at pages 77-80. Appellant's attorney

disagrees. At the indicated location, HotMetalPro merely describes inserting applets in an HTML page, but does not teach or suggest the above-identified limitations of Appellant's claims.

With regard to claims 9, 32 and 55, which recite mapping the entities defined in the schema to components of the editor, the Office Action asserts that these limitations are taught by HotMetalPro at pages 78-79. Appellant's attorney disagrees. At the indicated location, HotMetalPro merely describes the specification of applets, but does not teach or suggest the above-identified limitations of Appellant's claims.

With regard to claims 10, 33 and 56, which recite that the entities are selected from a group comprising elements and attributes of elements, the Office Action asserts that these limitations are taught by HotMetalPro at pages 78-79. Appellant's attorney disagrees. At the indicated location, HotMetalPro merely describes the specification of applets, but does not teach or suggest attributes in the same context as Appellant's claimed invention.

With regard to claims 11, 34 and 57, which recite that the attribute has a declaration selected from a group comprising mandatory, optional, and fixed value, the Office Action asserts that these limitations are taught by W3C at pages 18-21. Appellant's attorney disagrees. At the indicated location, W3C merely describes XML attribute types, but does not teach or suggest attributes in the same context as Appellant's claimed invention.

With regard to claims 12, 35 and 58, which recite accepting user input for attributes having a mandatory declaration, the Office Action asserts that these limitations are taught by W3C at pages 18-21. Appellant's attorney disagrees. At the indicated location, W3C merely describes XML attribute types, but does not teach or suggest attributes in the same context as Appellant's claimed invention.

With regard to claims 13, 36 and 59, which recite accepting user input for attributes having an optional declaration, the Office Action asserts that these limitations are taught by W3C at pages 18-21. Appellant's attorney disagrees. At the indicated location, W3C merely describes XML attribute types, but does not teach or suggest attributes in the same context as Appellant's claimed invention.

With regard to claims 14, 37 and 60, which recite entering values from the schema for attributes having a fixed value declaration, the Office Action asserts that these limitations are taught by W3C at pages 18-21. Appellant's attorney disagrees. At the indicated location, W3C merely

describes XML attribute types, but does not teach or suggest attributes in the same context as Appellant's claimed invention.

With regard to claims 15, 38 and 61, which recite validating values entered for the attribute, the Office Action asserts that these limitations are taught by W3C at page 31. Appellant's attorney disagrees. At the indicated location, W3C merely describes validating processors for XML documents, but does not teach or suggest validating values for attributes in the same context as Appellant's claimed invention.

With regard to claims 16, 39 and 62, which recite that the class specifications include at least one function for validating at least one entity defined in the schema, the Office Action asserts that these limitations are taught by W3C at page 31. Appellant's attorney disagrees. At the indicated location, W3C merely describes validating processors for XML documents, but does not teach or suggest the above-identified limitations of Appellant's claims.

With regard to claims 17, 40 and 63, which recite that the generating further comprises generating the class specifications from a regular expression language comprising one or more declarations of elements enclosed within an element, the Office Action asserts that these limitations are taught by Bray at page 5 and 38. Appellant's attorney disagrees. At the indicated location, Bray merely describes the grammar of XML, but does not teach or suggest the above-identified limitations of Appellant's claims.

With regard to claims 18, 41 and 64, which recite that the regular expression language includes one or more regular expression operators selected from a group comprising:

(1) a "zero or more" operator,

(2) a "one or more" operator,

(3) a "one or the other" operator,

(4) a "one followed by the other" operator,

(5) a "zero or one" operator,

(6) a "grouping" operator, and

(7) an "any" operator,

the Office Action asserts that these limitations are taught by Bray at page 5 and 38. Appellant's attorney disagrees. At the indicated location, Bray merely describes the grammar of XML, but does not teach or suggest the above-identified limitations of Appellant's claims.

With regard to claims 19, 42 and 65, which recite that the class specifications define one or more widgets that are associated with each of the operators, the Office Action asserts that these limitations are taught by Bray at page 5 and 38. Appellant's attorney disagrees. At the indicated location, Bray merely describes the grammar of XML, but does not teach or suggest the above-identified limitations of Appellant's claims.

With regard to claims 20, 43 and 66, which recite that the class specifications define at least one widget associated with an entity in the schema, the Office Action asserts that these limitations are taught by HotMetalPro at pages 78-79. Appellant's attorney disagrees. At the indicated location, HotMetalPro merely describes the specification of applets for use with HTML pages, but does not teach or suggest the above-identified limitations of Appellant's claims.

With regard to claims 21, 44 and 67, which recite identifying specific widget implementations for use with the editor, the Office Action asserts that these limitations are taught by HotMetalPro at pages 77-80. Appellant's attorney disagrees. At the indicated location, HotMetalPro merely describes inserting applets in an HTML page, but does not teach or suggest the above-identified limitations of Appellant's claims.

With regard to claims 22, 45 and 68, which recite customizing the editor for use with different regular expression operators, the Office Action asserts that these limitations are taught by HotMetalPro at pages 78-79. Appellant's attorney disagrees. At the indicated location, HotMetalPro merely describes the specification of applets for use with HTML pages, but does not teach or suggest the above-identified limitations of Appellant's claims.

With regard to claims 23, 46 and 69, which recite attempting to solve correctness, optimization, or aesthetics related issues when generating the visual editor from the schema, the Office Action asserts that these limitations are taught by W3C. Appellant's attorney disagrees. W3C merely describes the XML standard, but does not teach or suggest the above-identified limitations of Appellant's claims.

IX. CONCLUSION

In light of the above arguments, Appellant's attorney respectfully submits that the cited references do not anticipate nor render obvious the claimed invention. More specifically, Appellant's claims recite novel physical features which patentably distinguish over any and all

references under 35 U.S.C. §§ 102 and 103. As a result, a decision by the Board of Patent Appeals and Interferences reversing the Examiner and directing allowance of the pending claims in the subject application is respectfully solicited.

Respectfully submitted,

GATES & COOPER LLP
Attorneys for Appellant

Howard Hughes Center
6701 Center Drive West, Suite 1050
Los Angeles, California 90045
(310) 641-8797

Date: June 28, 2004

By: _____
Name: George H. Gates
Reg. No.: 33,500

GHG/amb
G&C 30879.71-US-01

APPENDIX

1. A computer-implement method for generating a document editor, comprising:

(a) generating one or more class specifications in the computer from a schema for the document, wherein the class specifications identify user interface components of the editor corresponding to entities defined in the schema; and

(b) instantiating one or more objects in the computer from the class specifications to invoke the editor.

2. The method of claim 1 above, wherein the documents are eXtensible Markup Language (XML) documents and the schemas are XML schemas.

3. The method of claim 2 above, wherein the schemas are selected from a group including Document Type Definition (DTD) schemas, Document Content Definition (DCD) schemas, and XSchema schemas.

4. The method of claim 1 above, wherein the class specifications comprise Java class specifications.

5. The method of claim 1 above, wherein the generating step further comprises converting an entity defined in the schema into the class specification.

6. The method of claim 1 above, wherein the generating step further comprises the step of generating the class specifications in the computer from the schemas and one or more optional customization specifications.

7. The method of claim 6 above, wherein the optional customization specifications define what class names to generate for each entity defined in the schema.

8. The method of claim 1 above, wherein the class specifications include one or more specifications selected from a group comprising (1) a visual editor class specification, (2) a content implementation class specification, and a handler class specification.

9. The method of claim 1 above, further comprising mapping the entities defined in the schema to components of the editor.

10. The method of claim 1 above, wherein the entities are selected from a group comprising elements and attributes of elements.

11. The method of claim 10 above, wherein the attribute has a declaration selected from a group comprising mandatory, optional, and fixed value.

12. The method of claim 11 above, further comprising accepting user input for attributes having a mandatory declaration.

13. The method of claim 11 above, further comprising accepting user input for attributes having an optional declaration.

14. The method of claim 11 above, further comprising entering values from the schema for attributes having a fixed value declaration.

15. The method of claim 10 above, further comprising validating values entered for the attribute.

16. The method of claim 1 above, wherein the class specifications include at least one function for validating at least one entity defined in the schema.

17. The method of claim 1 above, wherein the generating step further comprises the step of generating the class specifications from a regular expression language comprising one or more declarations of elements enclosed within an element.

18. The method of claim 17 above, wherein the regular expression language includes one or more regular expression operators selected from a group comprising:

(1) a "zero or more" operator,

(2) a "one or more" operator,

(3) a "one or the other" operator,

(4) a "one followed by the other" operator,

(5) a "zero or one" operator,

(6) a "grouping" operator, and

(7) an "any" operator.

19. The method of claim 18 above, wherein the class specifications define one or more widgets that are associated with each of the operators.

20. The method of claim 1 above, wherein the class specifications define at least one widget associated with an entity in the schema.

21. The method of claim 1 above, further comprising identifying specific widget implementations for use with the editor.

22. The method of claim 1 above, further comprising customizing the editor for use with different regular expression operators.

23. The method of claim 1 above, further comprising attempting to solve correctness, optimization, or aesthetics related issues when generating the visual editor from the schema.

24. A computer-implemented apparatus for generating a document editor, comprising:

(a) a computer; and

(b) an editor maker, executed by the computer, for generating one or more class specifications in the computer from a schema for the document, wherein the class specifications identify user interface components of the editor corresponding to entities defined in the schema, and for instantiating one or more objects in the computer from the class specifications to invoke the editor.

25. The apparatus of claim 24 above, wherein the documents are eXtensible Markup Language (XML) documents and the schemas are XML schemas.

26. The apparatus of claim 25 above, wherein the schemas are selected from a group including Document Type Definition (DTD) schemas, Document Content Definition (DCD) schemas, and XSchema schemas.

27. The apparatus of claim 24 above, wherein the class specifications comprise Java class specifications.

28. The apparatus of claim 24 above, wherein the means for generating further comprises means for converting an entity defined in the schema into the class specification.

29. The apparatus of claim 24 above, wherein the means for generating further comprises means for generating the class specifications in the computer from the schemas and one or more optional customization specifications.

30. The apparatus of claim 29 above, wherein the optional customization specifications define what class names to generate for each entity defined in the schema.

31. The apparatus of claim 24 above, wherein the class specifications include one or more specifications selected from a group comprising (1) a visual editor class specification, (2) a content implementation class specification, and a handler class specification.

32. The apparatus of claim 24 above, further comprising means for mapping the entities defined in the schema to components of the editor.

33. The apparatus of claim 24 above, wherein the entities are selected from a group comprising elements and attributes of elements.

34. The apparatus of claim 33 above, wherein the attribute has a declaration selected from a group comprising mandatory, optional, and fixed value.

35. The apparatus of claim 34 above, further comprising means for accepting user input for attributes having a mandatory declaration.

36. The apparatus of claim 34 above, further comprising means for accepting user input for attributes having an optional declaration.

37. The apparatus of claim 34 above, further comprising means for entering values from the schema for attributes having a fixed value declaration.

38. The apparatus of claim 33 above, further comprising means for validating values entered for the attribute.

39. The apparatus of claim 24 above, wherein the class specifications include at least one function for validating at least one entity defined in the schema.

40. The apparatus of claim 24 above, wherein the means for generating further comprises means for generating the class specifications from a regular expression language comprising one or more declarations of elements enclosed within an element.

41. The apparatus of claim 40 above, wherein the regular expression language includes one or more regular expression operators selected from a group comprising:

(1) a "zero or more" operator,

(2) a "one or more" operator,

(3) a "one or the other" operator,

(4) a "one followed by the other" operator,

(5) a "zero or one" operator,

(6) a "grouping" operator, and

(7) an "any" operator.

42. The apparatus of claim 41 above, wherein the class specifications define one or more widgets that are associated with each of the operators.

43. The apparatus of claim 24 above, wherein the class specifications define at least one widget associated with an entity in the schema.

44. The apparatus of claim 24 above, further comprising means for identifying specific widget implementations for use with the editor.

45. The apparatus of claim 24 above, further comprising means for customizing the editor for use with different regular expression operators.

46. The apparatus of claim 24 above, further comprising means for attempting to solve correctness, optimization, or aesthetics related issues when generating the visual editor from the schema.

47. An article of manufacture embodying logic for performing a method for generating a document editor for use in an object-oriented computer system, the method comprising the steps of:

(a) generating one or more class specifications from a schema for the document, wherein the class specifications identify user interface components of the editor corresponding to entities defined in the schema; and

(b) instantiating one or more objects from the class specifications to invoke the editor.

48. The method of claim 47 above, wherein the documents are eXtensible Markup Language (XML) documents and the schemas are XML schemas.

49. The method of claim 48 above, wherein the schemas are selected from a group including Document Type Definition (DTD) schemas, Document Content Definition (DCD) schemas, and XSchema schemas.

50. The method of claim 47 above, wherein the class specifications comprise Java class specifications.

51. The method of claim 47 above, wherein the generating step further comprises converting an entity defined in the schema into the class specification.

52. The method of claim 47 above, wherein the generating step further comprises the step of generating the class specifications in the computer from the schemas and one or more optional customization specifications.

53. The method of claim 52 above, wherein the optional customization specifications define what class names to generate for each entity defined in the schema.

54. The method of claim 47 above, wherein the class specifications include one or more specifications selected from a group comprising (1) a visual editor class specification, (2) a content implementation class specification, and a handler class specification.

55. The method of claim 47 above, further comprising mapping the entities defined in the schema to components of the editor.

56. The method of claim 47 above, wherein the entities are selected from a group comprising elements and attributes of elements.

57. The method of claim 56 above, wherein the attribute has a declaration selected from a group comprising mandatory, optional, and fixed value.

58. The method of claim 57 above, further comprising accepting user input for attributes having a mandatory declaration.

59. The method of claim 57 above, further comprising accepting user input for attributes having an optional declaration.

60. The method of claim 57 above, further comprising entering values from the schema for attributes having a fixed value declaration.

61. The method of claim 56 above, further comprising validating values entered for the attribute.

62. The method of claim 47 above, wherein the class specifications include at least one function for validating at least one entity defined in the schema.

63. The method of claim 47 above, wherein the generating step further comprises the step of generating the class specifications from a regular expression language comprising one or more declarations of elements enclosed within an element.

64. The method of claim 63 above, wherein the regular expression language includes one or more regular expression operators selected from a group comprising:

(1) a "zero or more" operator,

(2) a "one or more" operator,

(3) a "one or the other" operator,

(4) a "one followed by the other" operator,

(5) a "zero or one" operator,

(6) a "grouping" operator, and

(7) an "any" operator.

65. The method of claim 64 above, wherein the class specifications define one or more widgets that are associated with each of the operators.

66. The method of claim 47 above, wherein the class specifications define at least one widget associated with an entity in the schema.

67. The method of claim 47 above, further comprising identifying specific widget implementations for use with the editor.

68. The method of claim 47 above, further comprising customizing the editor for use with different regular expression operators.

69. The method of claim 47 above, further comprising attempting to solve correctness, optimization, or aesthetics related issues when generating the visual editor from the schema.